WHAT IS CLAIMED IS:

1. A method of handling memory read return data from different time domains, comprising:

determining a number of distinct memory device ranks;

determining a time domain for each of the distinct memory device ranks; and

scheduling a transaction based on the time domain for each of the distinct memory device ranks so that at least one of data collisions and out-of-order data returns are prevented.

2. The method according to claim 1, further including determining a relative position of each of the distinct memory device ranks.

3. The method according to claim 1, wherein the determining of the number of the distinct memory device ranks is performed utilizing Serial Presence Detect (SPD).

4. The method according to claim 1, wherein the determining of the time domain for each of the distinct memory device ranks includes:

writing a predetermined data pattern to a memory device rank to be tested;

reading back the predetermined data pattern;

receiving the predetermined data pattern, assuming that the time domain of the memory device rank to be tested is in a first time domain;

determining whether the predetermined data pattern was correctly received;

increasing the time domain of the memory device rank to be tested by at least a clock if the predetermined data pattern was not correctly received; and

establishing the time domain for the memory device rank to be tested once the predetermined data pattern is correctly received.

5. The method according to claim 1, wherein the scheduling of the transaction includes:

determining whether a new request is available;

determining whether there are pending or outstanding transactions if the new request is available;

consulting a history of pending or outstanding transactions;

determining whether a data contention conflict exists;

determining whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists;

waiting at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later;

determining whether an out-of-order data conflict exists if the data contention conflict does not exist, and scheduling the transaction if the out-of-order data conflict does not exist; and

determining if the out-of-order data conflict exists if the data contention conflict can be resolved by scheduling the transaction now and sending the transaction later, and scheduling the transaction if the out-of-order data conflict does not exist.

6. The method according to claim 5, further including:

determining whether out-of-order data conflicts are allowed if the out-of-order data conflict exists, and scheduling the transaction if out-of-order data conflicts are allowed;

determining whether the out-of-order data conflict may be resolved by scheduling the transaction now and sending the transaction later if out-of-order data conflicts are not allowed;

waiting at least a second clock if the out-of-order data conflict cannot be resolved by scheduling the transaction now and sending the transaction later; and

scheduling the transaction if the out-of-order data conflict may be resolved by scheduling the transaction now and sending the transaction later.

7. The method according to claim 1, wherein the scheduling of the transaction includes:

determining whether a new request is available;

determining whether there are pending or outstanding transactions if the new request is available;

consulting a history of pending or outstanding transactions;

determining whether a data contention conflict exists;

determining whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists;

waiting at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later;

scheduling the transaction if the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later; and

scheduling the transaction if the data contention conflict does not exist.

8. A memory system, comprising:

a plurality of distinct memory device ranks;

a memory controller having a connection with the plurality of the distinct memory device ranks, wherein the memory controller is adapted to determine a number of the distinct memory device ranks, to determine a time domain for each of the distinct memory device ranks, and to schedule a transaction based on the time domain for each of the distinct memory device ranks so that at least one of data collisions and out-of-order data returns are prevented.

9. The memory system according to claim 8, wherein the memory controller is further adapted to determine a relative position of each of the distinct memory device ranks.

10. The memory system according to claim 8, wherein the memory controller utilizes Serial Presence Detect (SPD) to determine the number of the distinct memory device ranks.

11. The memory system according to claim 8, wherein the memory controller, in order to determine the time domain for each of the distinct memory device ranks, is adapted to write a predetermined data pattern to a memory device rank to be tested, to read back the predetermined data pattern, to receive the predetermined data pattern assuming that the time

domain of the memory device rank to be tested is in a first time domain, to determine whether the predetermined data pattern was correctly received, to increase the time domain of the memory device rank to be tested by at least a clock if the predetermined data pattern was not correctly received, and to establish the time domain for the memory device rank to be tested once the predetermined data pattern is correctly received.

12. The memory system according to claim 8, wherein the memory controller, in order to schedule the transaction, is adapted to determine whether a new request is available, to determine whether there are pending or outstanding transactions if the new request is available, to consult a history of pending or outstanding transactions, to determine whether a data contention conflict exists, to determine whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists, waiting at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later, to determine if an out-of-order data conflict exists if the data contention conflict does not exist and schedule the transaction if the out-of-order data conflict does not exist, and to determine if the out-of-order data conflict exists if the data contention conflict can be resolved by scheduling the transaction now and sending the transaction later and schedule the transaction if the out-of-order data conflict does not exist.

13. The memory system according to claim 12, wherein the memory controller, in order to schedule the transaction, is further adapted to determine whether out-of-order data conflicts are allowed if the out-of-order data conflict exists and schedule the transaction if out-of-order data conflicts are allowed, to determine whether the out-of-order data conflict may be

resolved by scheduling the transaction now and sending the transaction later if out-of-order data conflicts are not allowed, to wait at least a second clock if the out-of-order data conflict cannot be resolved by scheduling the transaction now and sending the transaction later, and to schedule the transaction if the out-of-order data conflict may be resolved by scheduling the transaction now and sending the transaction later.

14. The memory system according to claim 8, wherein the memory controller, in order to schedule the transaction, is adapted to determine whether a new request is available, to determine whether there are pending or outstanding transactions if the new request is available, to consult a history of pending or outstanding transactions, to determine whether a data contention conflict exists, to determine whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists, waiting at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later, to schedule the transaction if the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later, and to schedule the transaction if the data contention conflict does not exist.

15. The memory system according to claim 8, wherein the connection is a bus.

16. The memory system according to claim 15, wherein the bus includes a data bus and an address/command bus.

17. A memory controller, comprising:

a machine-readable medium; and

machine-readable program code, stored on the machine-readable medium, having instructions to,

determine a number of distinct memory device ranks,

determine a time domain for each of the distinct memory device ranks, and

schedule a transaction based on the time domain for each of the distinct memory device ranks so that at least one of data collisions and out-of-order data returns are prevented.

18. The memory controller according to claim 17, wherein the machine-readable program code includes instructions to determine a relative position of each of the distinct memory device ranks.

19. The memory controller according to claim 17, wherein the memory controller utilizes Serial Presence Detect (SPD) to determine the number of the distinct memory device ranks.

20. The memory controller according to claim 17, wherein the machine-readable program code, to determine the time domain for each of the distinct memory device ranks, includes instructions to:

write a predetermined data pattern to a memory device rank to be tested;

read back the predetermined data pattern;

receive the predetermined data pattern, assuming that the time domain of the memory device rank to be tested is in a first time domain;

determine whether the predetermined data pattern was correctly received;

increase the time domain of the memory device rank to be tested by at least a clock if the predetermined data pattern was not correctly received; and

establish the time domain for the memory device rank to be tested once the predetermined data pattern is correctly received.

21. The memory controller according to claim 17, wherein the machine-readable program code, to schedule the transaction, includes instructions to:

determine whether a new request is available;

determine whether there are pending or outstanding transactions if the new request is available;

consult a history of pending or outstanding transactions;

determine whether a data contention conflict exists;

determine whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists;

wait at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later;

determine if an out-of-order data conflict exists if the data contention conflict does not exist, and scheduling the transaction if the out-of-order data conflict does not exist; and

determine whether the out-of-order data conflict exists if the data contention conflict can be resolved by scheduling the transaction now and sending the transaction later, and scheduling the transaction if the out-of-order data conflict does not exist.

22. The memory controller according to claim 17, wherein the machine-readable program code, to schedule the transaction, further includes instructions to:

determine whether out-of-order data conflicts are allowed if the out-of-order data conflict exists, and scheduling the transaction if out-of-order data conflicts are allowed;

determine whether the out-of-order data conflict may be resolved by scheduling the transaction now and sending the transaction later if out-of-order data conflicts are not allowed;

wait at least a second clock if the out-of-order data conflict cannot be resolved by scheduling the transaction now and sending the transaction later; and

schedule the transaction if the out-of-order data conflict may be resolved by scheduling the transaction now and sending the transaction later.

23. The memory controller according to claim 17, wherein the machine-readable program code, to schedule the transaction, includes instructions to:

determine whether a new request is available;

determine whether there are pending or outstanding transactions if the new request is available;

consult a history of pending or outstanding transactions;

determine whether a data contention conflict exists;

determine whether the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later if the data contention conflict exists;

wait at least a clock if the data contention conflict cannot be resolved by scheduling the transaction now and sending the transaction later;

schedule the transaction if the data contention conflict may be resolved by scheduling the transaction now and sending the transaction later; and

schedule the transaction if the data contention conflict does not exist.